Efficiency Modeling and Analysis of 64-bit ARM Clusters for HPC

Joel Wanza, Sébastien Bilavarn, Said Derradji, Cécile Belleudy, Sylvie Lesmanne

Efficiency Modeling and Analysis of 64-bit ARM Clusters for HPC

Joel Wanza[1], Sébastien Bilavarn[2], Said Derradji[1], Cécile Belleudy[2], Sylvie Lesmanne[1]

***Abstract*— This paper investigates the use of ARM 64-bit cores to improve the processing efficiency of upcoming HPC systems. It describes a set of available tools, models and platforms, and their combination in an efficient methodology for the design space exploration of large manycore computing clusters. Experimentation and results using representative benchmarks allow to set an exploration approach to evaluate essential design options at micro-architectural level while scaling with a large number of cores, and to envisage first directions for future system analysis and improvement.**

I. Introduction

The performance of supercomputers has traditionally grown continuously with the advances of Moore's law and parallel processing, while energy efficiency could be considered as a secondary problem. But it quickly became clear that power consumption was the dominant term in the scaling challenges to reach the next level. It is roughly considered that 20 times energy efficiency improvements is required for exascale computing (10^{18} FLOPS) to cope with the tremendous electrical power and cost incurred by such computational capacity. The idea of using concepts borrowed from embedded technologies has naturally emerged to address this. First prototypes based on large numbers of low power manycore microprocessors (possibly millions of cores) instead of fast complex cores started to be investigated, putting forward a number of proposals for improvement at node level architecture to meet HPC demands. The recent advances in this area, namely the emergence and availability of 64-bit ARM cores, opens up new promising expectations that can start to be investigated. This paper is one of the first to describe outcomes of research in this field, with the analysis of available tools, models and platforms that can be efficiently used to explore the architecture design space of large manycore clusters based on the ARMv8 ISA.

The outline of the paper is the following. First, we present the context of this work and a state of the art on previous contributions related to the use of ARM cores for energy efficient HPC. In section 3, different modeling and simulation tools are examined in the context of realistic HPC benchmarking using lately available ARM 64-bit based platforms (ARM Juno, AMD Seattle, AppliedMicro X-Gene). Section 4 provides then extensive estimation versus measurement results that are further analyzed in detail to derive relevant HPC exploration methodology from the possible combination of tools and models. Section 5 presents asumptions for further design space exploration from this flow and section 6 exposes the main conclusions and next directions of research.

[1]Bull atos technologies
[2]LEAT - University of Nice Sophia Antipolis, CNRS

II. Context and previous works

Many previous works investigated the use of embedded processors to improve on the processing and energy efficiency of HPC systems. They covered a variety of 32-bit RISC cores ranging from ARM Cortex-A8 [1] and Cortex-A9 [2][3][4] to more recently Cortex-A15 and Cortex-A7 cores [5]. [2][3] and [4] addressed for example dual and quad core systems based on ARM Cortex-A9 cores. The different results indicated various processing limitations to meet HPC performance requirements, in terms of double precision floating point arithmetic, 32-bit memory controllers (limiting the address space), ECC memory (e.g. for scientific and financial computing), and fast interconnect (communication intensive applications). [6] and [7] additionaly confirmed that the variability in performance and energy could largely be attributed to floating point and SIMD computations, and interactions with the memory subsystem. Other works which addressed explicit comparison against x86 based systems also pointed out the need for higher levels of performance to meet HPC demands. [4] concludes that the cost advantage of ARM clusters diminishes progressively for computation-intensive applications (i.e. dynamic Web server application, video transcoding), and other works like [8] conducted on ARM Cortex-A8, Cortex-A9, Intel Sandybridge, and an Intel Atom confirmed that ARM and x86 could achieve similar energy efficiency, depending on the suitability of a workload to the microarchitectural features at core level.

Of the works addressing the feasibility of ARM SoCs based HPC systems, efforts focused widely on single-node performance using microbenchmarks. Less studies considered large-scale systems exceeding a few cores even though multi-node cluster performance is an essential aspect of future Exascale systems [9]. Considering further that new generations of cores such as the ARMv8-A ISA support features to improve specifically on HPC workloads (64-bit address space, 64-bit arithmetic, high speed interconnects, fast memory hierarchies), we address in this paper a study focussing on these issues. Therefore we provide an evaluation of available tools, models and platforms able to set the foundations of a methodical system level exploration approach for HPC applications scaling up to 128 ARM 64-bit cores.

III. Problem modeling

We first examine the use of available tools, models and platforms matching our needs. We then characterize a set of relevant HPC benchmarks on different platform configurations to verify that we meet all conditions for exploration effectiveness, given a set of architectural requirements to

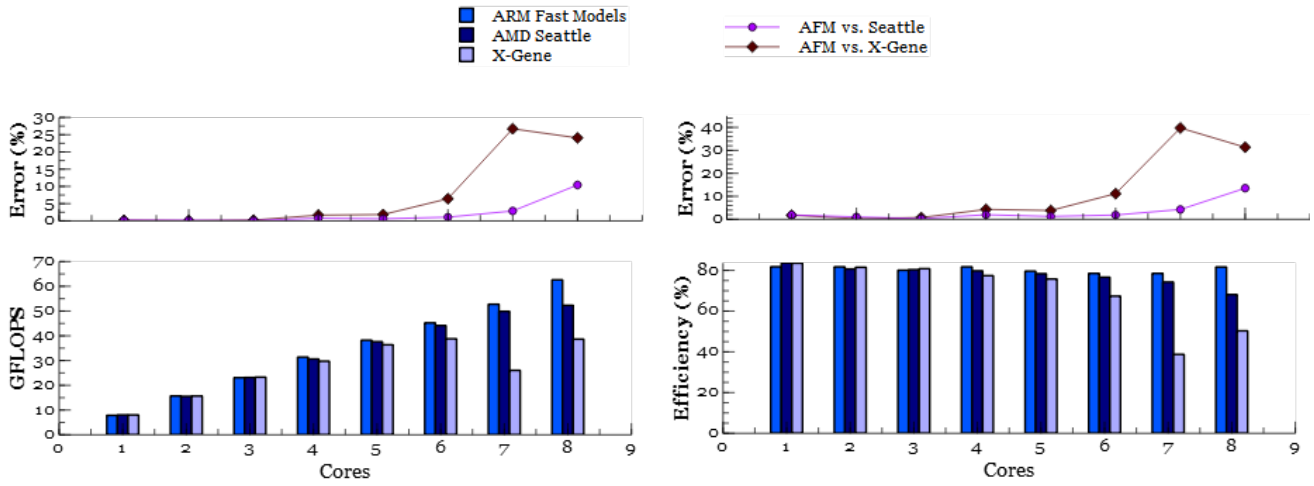


Fig. 1. Performance and efficiency of ARM Fast models vs. AMD Seattle and X-Gene platforms on SGEMM benchmark.

consider (performance, memory architecture, interconnect, scalability).

A. ARMv8 platforms

1) Juno ARM board: is a ARM development platform. Our interest goes mainly for the processor (two cortex-A57), the Cache Coherent Interconnect (CCI-400) and the DDR3-1600 dual channel memory controller.

2) AMD Seattle board: includes four clusters of two Cortex-A57 cores, with AMD Coherent Interconnect at 2GHz and two DDR4-3733 memory controllers.

3) AppliedMicro X-Gene1: Four clusters of two ARMv8 64-bit cores running at 2.4 GHz, AppliedMicro (APM) coherent network Interconnect and DDR 3 controller (16GB).

4) ARM Fast Model virtual platforms (AFM): It is the largest platform that can be configured with only ARM available fast model IPs, up to 48 Cortex-A57/A72. The limitation to 48 cores comes from the Cache Coherent Network CCN-512 interconnect supporting a maximum of 12 coherent clusters while each cluster can contain up to four cores.

5) Virtual Processing Unit (VPU): This is a synopsys methodology based platform, which enable to model efficiently extended flexible multicore SoC platforms in SystemC/TLM modeling language integrating several configurations without usual software constraints by using an interactive traffic called 'task graph' generated from the software traces executed on the AFM platform. We exploit this methodology mainly for a large scale ARM SoC coherent interconnect and memory sub-system architectural explorations.

Both the VPU and AFM platforms, are created with same characteristics than a corresponding real board for the correlation studies, in terms of frequency, the workload CPI (Cycle Per Instruction), memory bandwidth, caches hierarchy and sizes, etc.

B. Modeling tools

1) Platform Architect (AFM, VPU): The AFM and VPU platforms are created and used with this tool for Performance and Power Analysis. It has the advantage of both in the same tool, the critique of existing ARM technologies and an approach to architectural studies to improve with either type of complementary platforms AFM and VPU.

2) GEM5: is an academic and open source trace driven simulation tool, we use for cycle accurate full simulation (booting linux) of ARMv8 aarch64 processors.

C. Applications

Our floating point benchmarking is based on SGEMM, DGEMM [10] and HPL [11]. SGEMM and DGEMM measure the floating point rate of execution of respectively single precision and double precision real matrix-matrix multiplication, while HPL measures the floating point rate of execution for solving a linear system of equations. These benchmarks are commonly used in practice to help characterize system performances in terms of floating point operations. While by the other hand, we use the STREAM (Sustainable Memory Bandwidth in High Performance Computers [12]) benchmark to measure the memory controller bandwidth with four types of operations :

Functions Operations
Copy a(i) = b(i)
Scale a(i) = q*b(i)
Sum a(i) = b(i) + c(i)
Triad a(i) = b(i) + q*c(i)
Mean = (Copy + Scale + Sum + Triad) / 4

IV. Benchmarking analysis

We analyze the relevance of models and tools against real platforms (up to eight cores), firstly in terms of floating point processing performance and efficiency. We then focus on the memory and cache architecture, analyze the conditions of validity of the results, and extend the methodology to support robust analysis for a larger number of cores (possibly up to 128).

A. Performance models

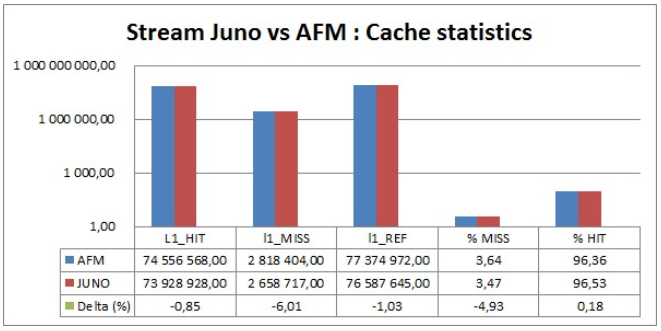
We consider two metrics to evaluate the processing efficiency. The first one is based on floating point operation per second (GFLOPS) which is reflective of the processing power for HPC workloads, and the second is the FLOPS efficiency expressing the ratio of actual versus theoretical FLOPS supported by the system. ARM Fast models are used as one objective is to examine the organization of efficient clusters, which can well benefit here from an ARM CoreLink CCN-512 interconnect model supporting up to twelve clusters of four A57 cores. Two real platforms (AMD Seattle, AppliedMicro X-Gene) supporting both four clusters of two ARMv8 64-bit cores with their built-in interconnect are thus used to compare the models with reality as reported in figure 1. These two real platforms are thus modeled in the virtual platform using A57 AFM models for all cores and the CCN-512 interconnect model in the absence of interconnect models for the AMD and AppliedMicro platforms.

The results indicate an average GFLOPS and efficiency accuracy of respectively 1.1% and 2.5% up to six cores. Then the disparity of interconnects on the different platforms reflects in deviations that are higly sensitive with the growing number of cores. The results show therefore that the global accuracy of AFM based virtual platforms is very good with less than 1.8% in average using ARM Fast models, but greatly dependent on the relevance of the interconnect model in configurations exceeding six cores. However, simulation times further limit the use of this platform in complex configurations (twenty two cores requires two days on a desktop workstation). This model can therefore be useful to explore core level, interconnect and intra cluster configuration. Further scalability will thus be addressed in another way as depicted in section IV-C and memory hierarchy is addressed in the following section.

B. Memory and cache architecture

The goal here is to extend previous approach to let the robust analysis of memory hierarchy performance (execution time and throughput) and performance scalability (considering the possible impacts of cache). Given previous outcome, these metrics and more especially the cache statistics only relates to the cluster level (L1 and L2 cache) and more importantly to the L1 cache which has the most performance impact and should typically have a hit rate above 95% in real world applications.

The Juno ARM platform gives access to advanced performance monitoring features of A57 and A53 cores. We can therefore configure and experiment with AFM platforms based on A57 cores to examine the precision of memory models against the Juno platform. The following



	L1_HIT	I1_MISS	I1_REF	% MISS	% HIT
AFM	74 556 568,00	2 818 404,00	77 374 972,00	3,64	96,36
JUNO	73 928 928,00	2 658 717,00	76 587 645,00	3,47	96,53
Delta (%)	-0,85	-6,01	-1,03	-4,93	0,18

Fig. 2. L1 cache statistics

thus describes simulation of a Cortex-A57 core running the STREAM benchmark where the results (figure 2) are plotted against the Juno A57 core in terms of cache statistics (L1 miss, L1 hit) for the STREAM benchmark with an average precision of $2,6\%$.

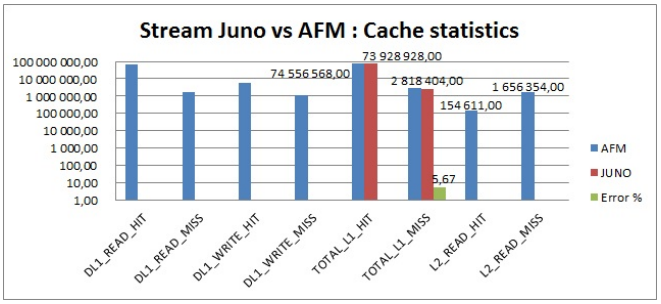
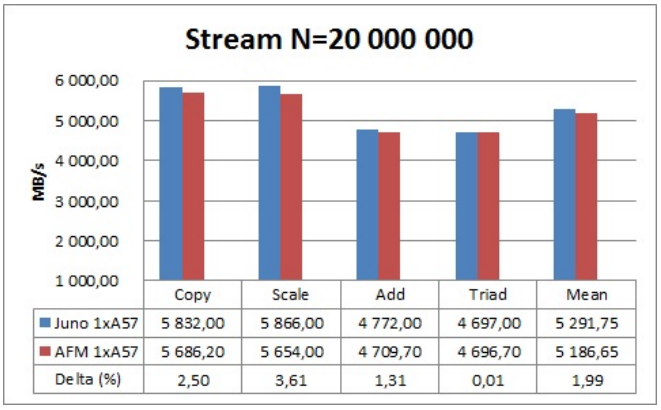


Fig. 3. L1 vs. L2 cache statistics

Figure 3 confirms that the L1 cache statistics are more meaningful than L2, which is logical because L2 cache traffic comes essentially from L1 misses of the A57 core used here. We can also observe that the cache miss correlation error doesn't exceed 6%.



	Copy	Scale	Add	Triad	Mean
Juno 1xA57	5 832,00	5 866,00	4 772,00	4 697,00	5 291,75
AFM 1xA57	5 686,20	5 654,00	4 709,70	4 696,70	5 186,65
Delta (%)	2,50	3,61	1,31	0,01	1,99

Fig. 4. Throughtput performance.

It is not possible to compare AFM platforms against real numbers in configurations exceeding two cores since only

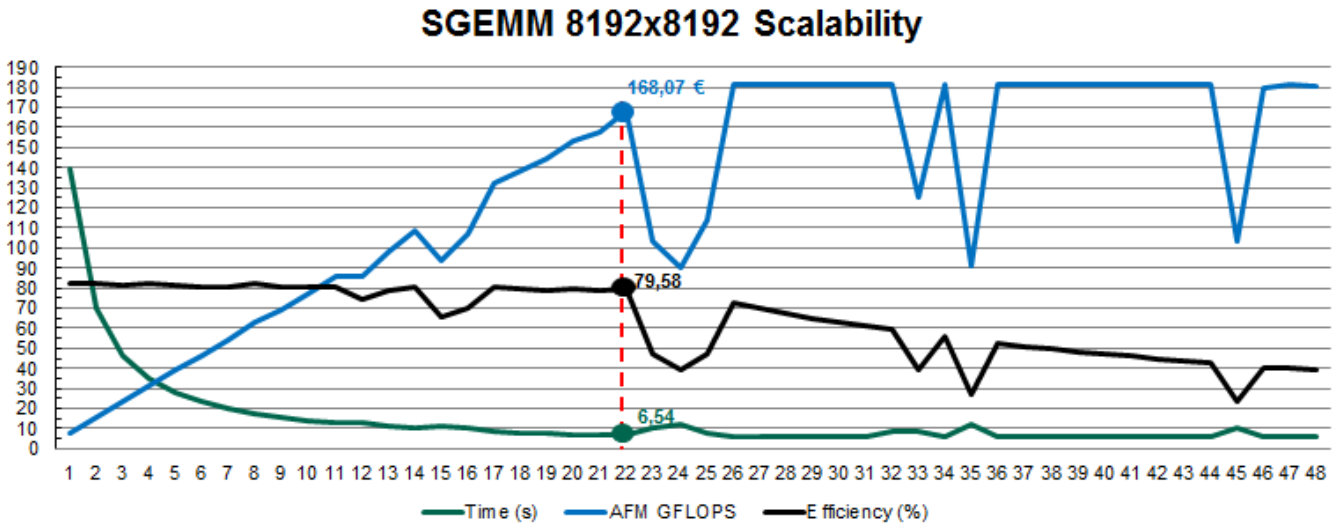


Fig. 5. Scalability on SGEMM benchmark.

two A57 are available on the Juno platform. Anyway as pointed out previously, growing deviation is likely to occur due to the difference between interconnects. However, on the 1xA57 reference configuration split over the five STREAM functions operations, AFM provides pinpoint accuracy with $1,9\%$ in throughput performance on average (figure 4).

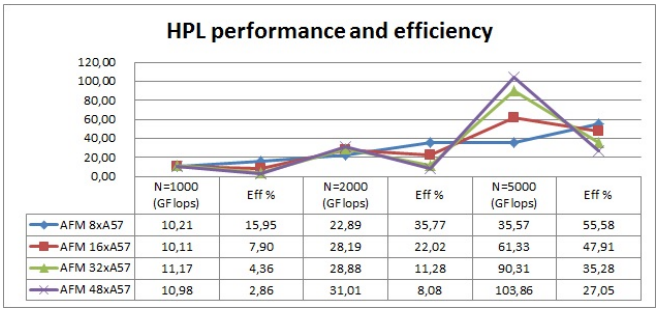
Therefore, this setup can be profitably used to identify improvement opportunities at the node/cluster level concerning the effect of different cache configurations (size, policy, topology) on system performances (to be discussed in section V).

C. Medium scale simulation

The scope of this part is to extend the analysis at a larger scale with AFM fast models available IPs. Due to CCN-512 limitation, we target configurations up to 48 cores in the following simulations (figure 5 and 6).

Figure 5 reports performance and efficiency analysis of the SGEMM benchmark executed on the AFM platform. While increasing the number of threads from 1 to 48, each thread is bound to an independent virtual Cortex-A57 core. Inspecting the Time and GFLOPS traces, system performance increases until the 22nd thread and then drops, indicating a peak for a 8192*8192 configuration (involving 1.5GB of RAM). This means that beyond this peak value, increasing the number of cores is useless for this benchmark configuration. A Larger SGEMM matrix size would be required to reach the peak at the 48th thread, but we start to exceed here the limits of AFM modeling abstraction level leading to prohibitive simulation times (more than 2 weeks).

Figure 6 reports performance and efficiency analysis of the HPL benchmark using larger AFM platforms configured for 8, 16, 32 and 48 threads. FLOPS efficiency increases gradually with parameter N. Optimized ATLAS libraries (Automatically Tuned Linar Algebra Software) are used in a way to reach the peak performance for 48 cores. However, larger values for N are needed to prevent the system from being under-used as visible in the results. Again this has not been further investigated because of excessive simulation times, but in spite of this, the available simulations provide valuable feedback in terms of possible hardware and software co-design analysis of the system. The loss of efficiency



	N=1000 (GFlops)	Eff %	N=2000 (GFlops)	Eff %	N=5000 (GFlops)	Eff %
AFM 8xA57	10,21	15,95	22,89	35,77	35,57	55,58
AFM 16xA57	10,11	7,90	28,19	22,02	61,33	47,91
AFM 32xA57	11,17	4,36	28,88	11,28	90,31	35,28
AFM 48xA57	10,98	2,86	31,01	8,08	103,86	27,05

Fig. 6. Scalability on HPL benchmark with BLAS optimized librairies.

observed for $N = 5000$ is explained simply by the fact that the theoretical performance increases linearly in proportion to the number of cores while the performance achieved is not the peak of HPL benchmark with optimized parameter configurations.

V. Architecture Exploration Perspectives

Previous section has led to develop a two-step exploration procedure that allows to assess architectural speculations at node/cluster level and medium scale analysis that can support up to 12 clusters of 4 cores (current limitation of the AFM platform). Larger scale analysis exceeding 128 cores is possible with the VPU platform supporting GCCI (Generic Cache Coherent Interconnect) Synopsys SystemC/TLM interconnect model while the fast model CCN-512 cannot hold more than 48 ARM coherent cores (section III-A).

On this basis, it is possible to address different topologies such as one large HPC specific SoC, several small SoCs on an interposer, several small SoCs on a System-in-Package.

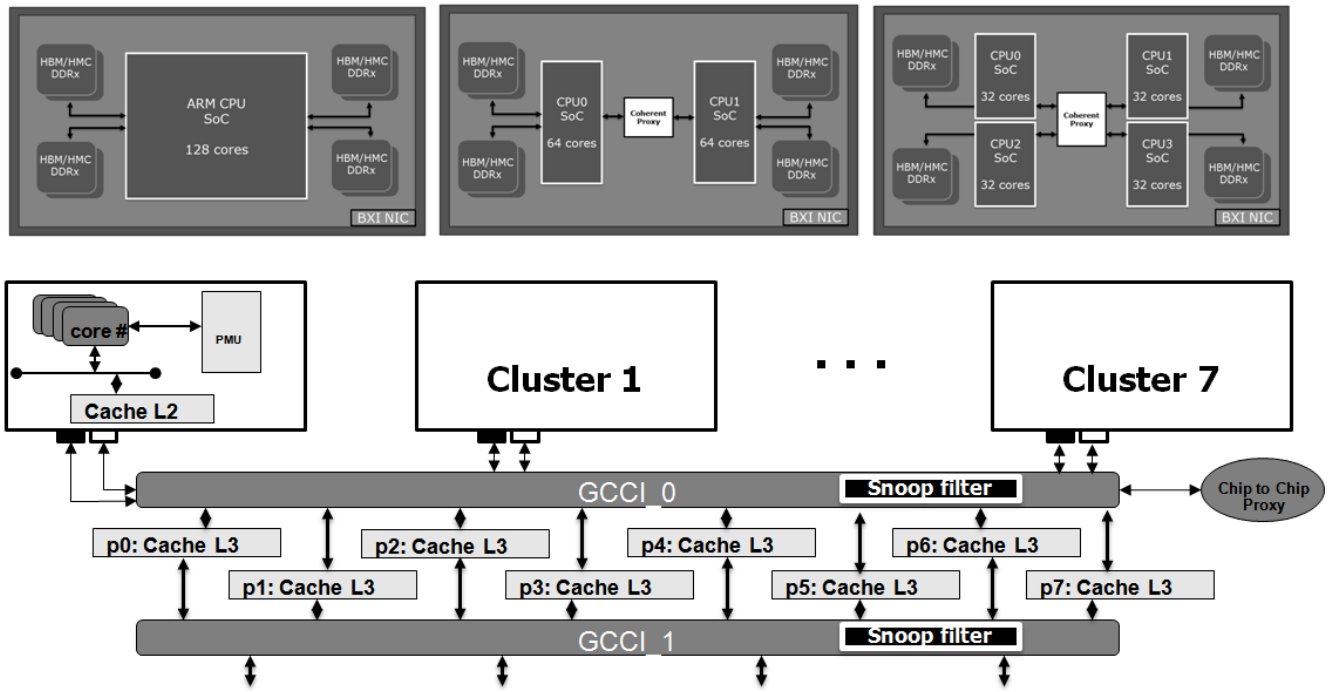


Fig. 7. Architecture exploration block diagram.

At the cluster level (top of figure 7), 1x128, 2x64, 4x32 are interesting organizations that can be addressed. As we aim at considering also the impact on energy efficiency, an investigation of power models is also underway to expand the scope of the approach to the analysis and improvement of the performance per watt. Power modeling and estimation included in the virtual platforms are based on state machines where the total power is computed from the relevant power of each component (caches, buses, VPU, etc) that are monitored at runtime. Additionally, ethernet or HPC clusters of these nodes will be described using Gem5-dist to help the collaboration of partners addressing different aspects of the global design space exploration,

VI. Conclusion

In this paper, we have examined in detail how a combined use of relevant models, tools, platforms and benchmarks could be used to define a robust design space exploration approach adapted to the tight processing efficiency constraints of upcoming HPC, especially in the new perspectives offered by ARMv8 64-bit cores. Proper architectural exploration is decomposed into two steps that allow i) reliable modeling and simulation at node/cluster level and ii) scalability analysis of a medium number of nodes using ARMv8 core models. Reported experiments and results have shown the ability of the approach to reliably study central design parameters, namely in terms of FLOPS performance and efficiency, cache and memory hierarchy, and scalability support up to 48 nodes. Further scalability and evaluation of power models is currently ongoing to extend the existing approach to support fine analysis for drastic improvements of energy efficiency. From there, future works will be able to focus on the evaluation and comprehensive analysis of diverse micro-achitectural opportunities at node, cluster, memory herarchy, interconnect and scalability levels.

Acknowledgment

This work is supported partly by the H2020 Mont-Blanc project and a French ANRT CIFRE partnership between Bull atos technologies and LEAT (University of Nice Sophia Antipolis, CNRS).

References

[1] K. Furlinger, C. Klausecker, and D. Kranzlmuller, Towards energy effcient parallel computing on consumer electronic devices, In Information and Communication on Technology for the Fight against Global Warming, pages 1-9. Springer, 2011.

[2] Edson L. Padoin, Daniel A. G. de Oliveira, Pedro Velho, Philippe O. A. Navaux, Brice Videau, Augustin Degomme, Jean-Francois Mehaut, Scalability and Energy Efficiency of HPC cluster with ARM MPSoC, 11th Workshop on Parallel and Distributed Processing (WSPPD), 2013.

[3] N. Rajovic, A. Rico, J. Vipond, I. Gelado, N. Puzovik, and A. Ramirez, Experiences with mobile processors for energy efficient hpc. Design, Automation and Test in Europe Conference and Exhibition (DATE), 2013.

[4] Z. Ou, B. Pang, Y. Deng, J. K. Nurminen, A. Yla-Jaaski, and P. Hui, Energy and cost-effciency analysis of ARM based clusters, Symposium on Cluster, Cloud and Grid Computing (CCGRID), 2012.

[5] Michael F. Cloutier, Chad Paradis and Vincent M. Weaver, Design and Analysis of a 32-bit Embedded High-Performance Cluster Optimized for Energy and Performance, 2014 Co-HPC Workshop.

[6] Michael A. Laurenzano, Ananta Tiwari, Adam Jundt, Joshua Peraza, William A. Ward, Jr., Roy Campbell, and Laura Carrington, Characterizing the Performance-Energy Tradeoff of Small ARM Cores in HPC Computation, Euro-Par 2014 Parallel Processing, Volume 8632 of the series Lecture Notes in Computer Science pp 124-137.

[7] Jahanzeb Maqbool, Sangyoon Oh, Geoffrey C. Fox, Evaluating Energy Efficient HPC Clusters for Scientific Workloads, Technical report, 2014.

[8] E. R. Blem, J. Menon, and K. Sankaralingam. Power struggles: Revisiting the risc vs. cisc debate on contemporary arm and x86 architectures. In HPCA, pages 1-12, 2013.

[9] A. Bhatele, P. Jetley, H. Gahvari, L. Wesolowski, W. D. Gropp, L. Kale, Architectural constraints to attain 1 exaflop/s for three scientific application classes, Proceedings of IEEE International Parallel and Distributed Processing Symposium, 2011, pp. 8091.

[10] C. L. Lawson, R. J. Hanson, D. Kincaid, F. T. Krogh. Basic Linear Algebra Subprograms for FORTRAN usage, ACM Trans. Math. Software 5: 308323. doi:10.1145/355841.355847. Algorithm 539, 1979.

[11] J. Dongarra, P. Luszczek, A. Petitet, The LINPACK Benchmark: Past, Present, and Future, Concurrency: Practice and Experience, 15, pp. 803-820, 2003.

[12] John D. McCalpin. Sustainable memory bandwidth in current high performance computers. Technical report, 1995.